GRAN SASSO
SCIENCE INSTITUTE

Sardinia site studies and characterization

Hands on session

Matteo Di Giovanni

INTRODUCTION

The aim of this hands on session is to present the softwares and libraries used so far for Sardinia site characterization.

All the procedures shown in this session are those used for the recent SRL publication about Sos Enattos.

The aim of the shared jupyter notebooks and Python scripts is to give the possibility to replicate the results and improve the procedures.

RESEARCH ARTICLE | NOVEMBER 04, 2020

A Seismological Study of the Sos Enattos Area—the Sardinia Candidate Site for the Einstein Telescope

Matteo Di Giovanni; Carlo Giunchi; Gilberto Saccorotti; Andrea Berbellini; Lapo Boschi; Marco Olivieri; Rosario De Rosa; Luca Naticchioni; Giacomo Oggiano; Massimo Carpinelli; Domenico D'Urso; Stefano Cuccuru; Valeria Sipala; Enrico Calloni; Luciano Di Fiore; Aniello Grado; Carlo Migoni; Alessandro Cardini; Federico Paoletti; Irene Fiori; Jan Harms; Ettore Majorana; Piero Rapagnani; Fulvio Ricci; Michele Punturo

Seismological Research Letters (2021) 92 (1): 352–364.

https://doi.org/10.1785/0220200186 Article history

Cite Share Tools

Abstract

The recent discovery of gravitational waves (GWs) and their potential for cosmic observations prompted the design of the future third-generation GW interferometers, able to extend the observation distance for sources up to the frontier of the Universe. In particular, the European detector Einstein Telescope (ET) has been proposed to reach peak strain sensitivities of about $3 \times 10^{-25}\ \mathrm{Hz}^{-1/2}$ in the 100 Hz frequency region and to extend the detection band down to 1 Hz. In the bandwidth [1,10] Hz, the seismic ambient noise is expected to represent the major perturbation to interferometric measurements, and the site that will host the future detectors must fulfill stringent requirements on seismic disturbances. In this article, we conduct a seismological study at the Italian ET candidate site, the dismissed mine of Sos Enattos in Sardinia. In the range between few mHz to hundreds of mHz, out of the detection bandwidth for ET, the seismic noise is compatible with the new low-noise model (Peterson, 1993); in the [0.1,1] Hz bandwidth, we found that seismic noise is correlated with sea wave height in the northwestern Mediterranean Sea. In the [1,10] Hz frequency band, noise is mainly due to anthropic activities; within the mine tunnels ($\simeq 100$ m underground), its spectrum is compliant with the requirements of the ET design. Noise amplitude decay with depth is consistent with a dominance of Rayleigh waves, as suggested by synthetic seismograms calculated for a realistic velocity structure obtained from the inversion of phase- and group-velocity dispersion data from array recording of a mine blasting. Further investigations are planned for a quantitative assessment of the principal noise sources and their spatiotemporal variations.

G S S I

INTRODUCTION

You can download the shared folder with scripts and notebooks from Google Drive by scanning the following QR code:

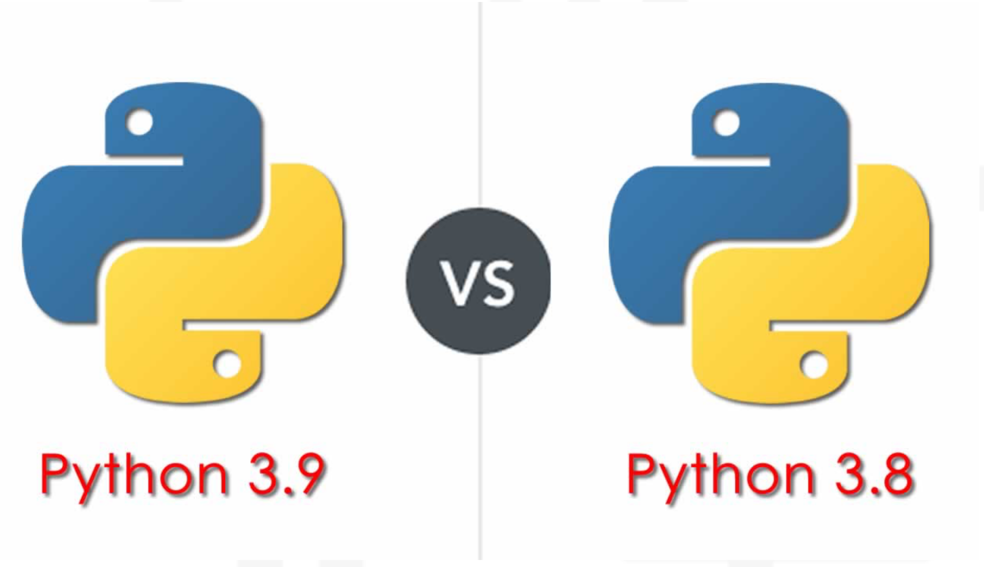
TIPS FOR INSTALLATION

When installing new versions of Python or new modules, my advice is not to mess with your native Python installation but to use Anaconda instead.

Anaconda is a virtual environment (venvs) package manager that lets you create venvs with any available Python verion.

For downloads and tutorials: www.anaconda.com

TIPS FOR INSTALLATION



When you create a Python venv, my advice is to not use the latest Python version available (3.9) but to stick to older releases like 3.7 or 3.8

Python modules may not be up to date with the most recent version and can create conflicts.

GSSI

TIPS FOR INSTALLATION

Install the ever-present Python modules:

- numpy
- scipy
- matplotlib

plus:

- obspy
- netCDF4
- xarray
- basemap
- motuclient

If you use Anaconda, it is enough to type `conda install module_name` in your venv.
Anaconda delivers 99% of available Python modules.

To simplify things, follow the installation instructions for ObsPy at https://github.com/obspy/obspy/wiki

NumPy

SciPy

matplotlib

ObsPy

A Python Framework for Seismology

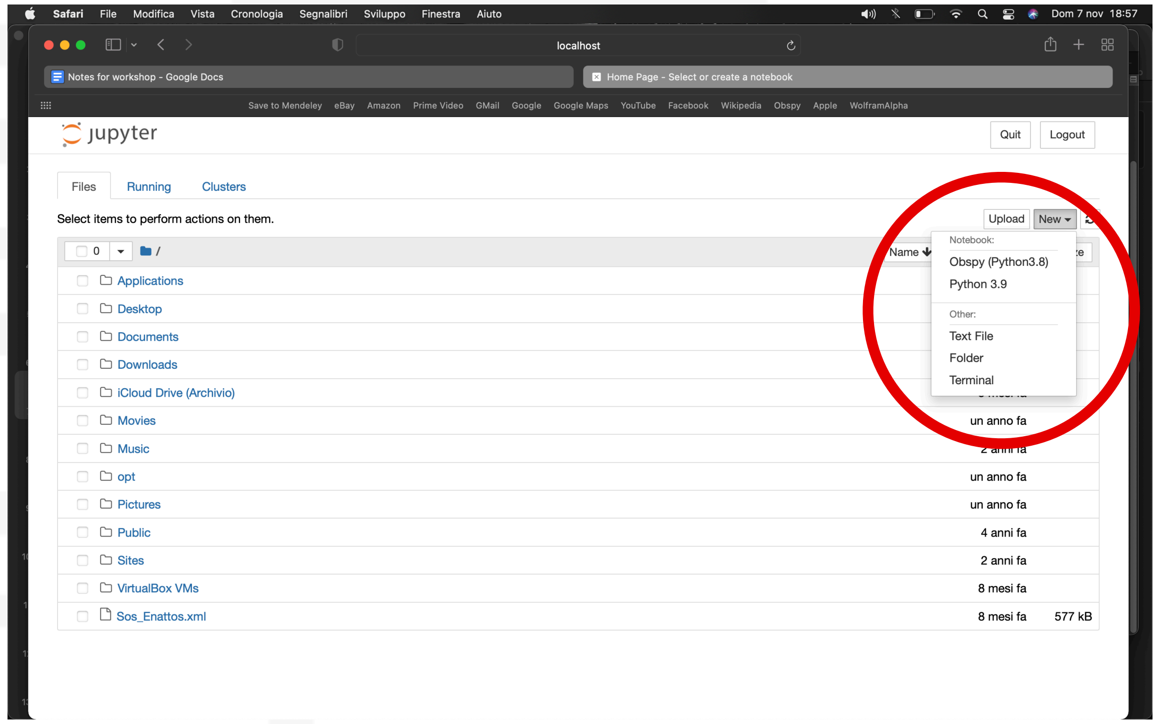
GSSI

TIPS FOR INSTALLATION

Since some of the provided scripts are Python notebooks, be sure to have Jupyter installed on your computer (https://jupyter.org/install).

Using iPykernel you can add as many kernels as you want to your Jupyter notebooks, including Anaconda kernels.

```
python -m ipykernel install --name venv_name
```



ObsPy

Obspy is a convenient Python module to read, save and manipulate seismic data. It provides lots of useful built-in functions that help facilitate the analysis of seismic data.

ObsPy

A Python Framework for Seismology

```
>conda config --add channels conda-forge
>conda create -n obspy python=3.7
> conda activate obspy
(obspy)> conda install obspy
```

Check the success of the installation by running Python in the venv and importing the obspy module.

G S
S I

Reading and saving data with ObsPy

- Obspy is compatible with the most common seismic data file formats (.sac, .sgy, .mseed…);
- The notebook `read_and_save_with_obspy.ipynb` provides examples of the various ways in which you can read and save seismic data.

Reading and saving data with ObsPy

- Obspy can read data from local and remote locations;
- Keep in mind that, on local directories, Obspy can read data using different procedures:
    - `read(file_name)` function;
    - SDS protocol;

GS
SI

Use of the `read` function

- Useful for small data chunks;
- Saves all the data (traces, times, ecc…) in a stream object.

SDS protocol

- Basic level of standardization and portability to softwares that need direct access to data files;
- Given data from a seismic station from a given year and characterized by the codes `NETWORK`, `STATION`, `LOCATION`, `CHANNEL`, the basic directory and file layouts are defined as:

```
PATH_TO_SDS/Year/NETWORK/STATION/CHANNEL.TYPE/NETWORK.STATION.LOCATION.CHANNEL.TYPE.YEAR.DAY
```

Stream object

Streams are list-like objects which contain multiple Trace objects, i.e. gap-less continuous time series and related header/meta information.

Each Trace object has an attribute called data pointing to a NumPy ndarray of the actual time series and the attribute stats which contains all meta information in a dictionary-like Stats object. Both attributes starttime and endtime of the Stats object are UTCDateTime objects.

Read files from remote servers

ObsPy makes use of the IRIS FDSN webservice to access publicly available seismic data.

The user must know the IRIS client name of the data provider (e.g. INGV), the network (e.g. IV) and station name (e.g. SENA).

To print all available IRIS clients, type the following:

```
from obspy.clients.fdsn.header import URL_MAPPINGS

for key in sorted(URL_MAPPINGS.keys()):
   print("{0:<11} {1}".format(key,  URL_MAPPINGS[key]))
```

Inventory files

Raw seismic data must be deconvoluted with the station's instrument response. This is done with the use of inventory files (often called dataless files) saved in various formats (.xml, .see, etc…).

```
inventory = read_inventory(file_name.xml)

inv = client.get_stations(network=net, station=sta,
location=loc, channel=cha, level="response")

stream.remove_response(inv = inventory, response = 'ACC')
```

G S
S I

Hands-on material

See the notebook `read_and_save_with_obspy.ipynb` *for examples on how to read and save data with ObsPy*

Use the notebook `create_inventory_file.ipynb` *to create your own inventory*

Spectra

Spectra can be calculated in many different ways using an almost infinite variety of functions. I focused on two methods for the analysis:

- ObsPy PPSD function;
- SciPy fft/spectrogram functions

ObsPy

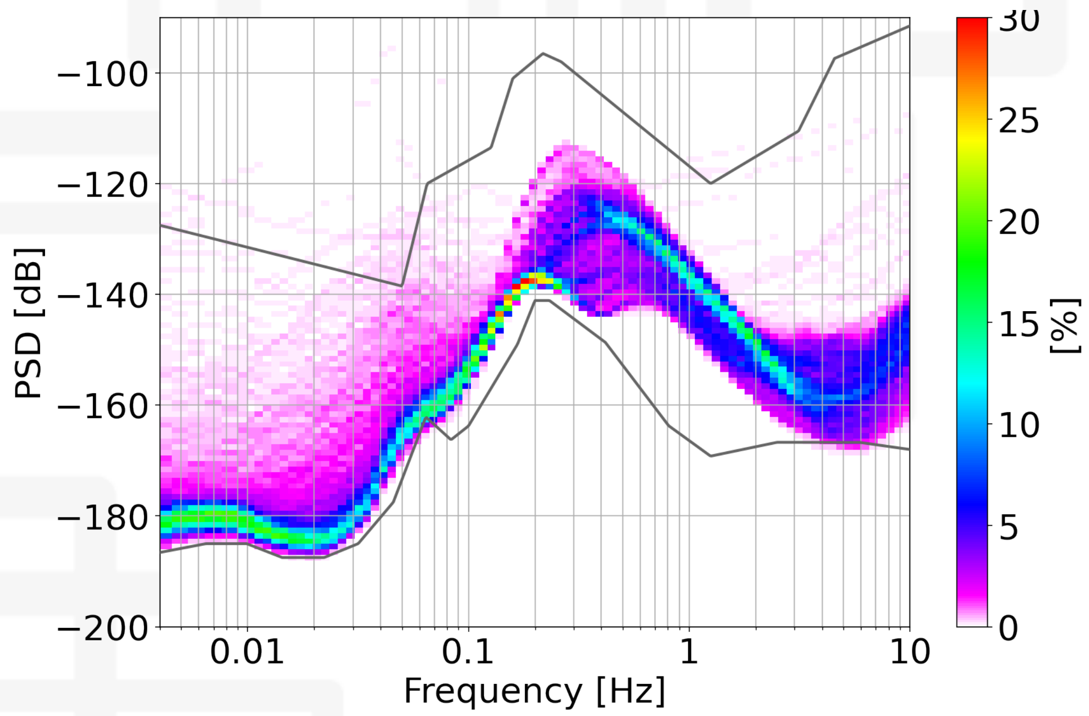
A Python Framework for Seismology

vs.

SciPy

G S
S I

ObsPy PPSD



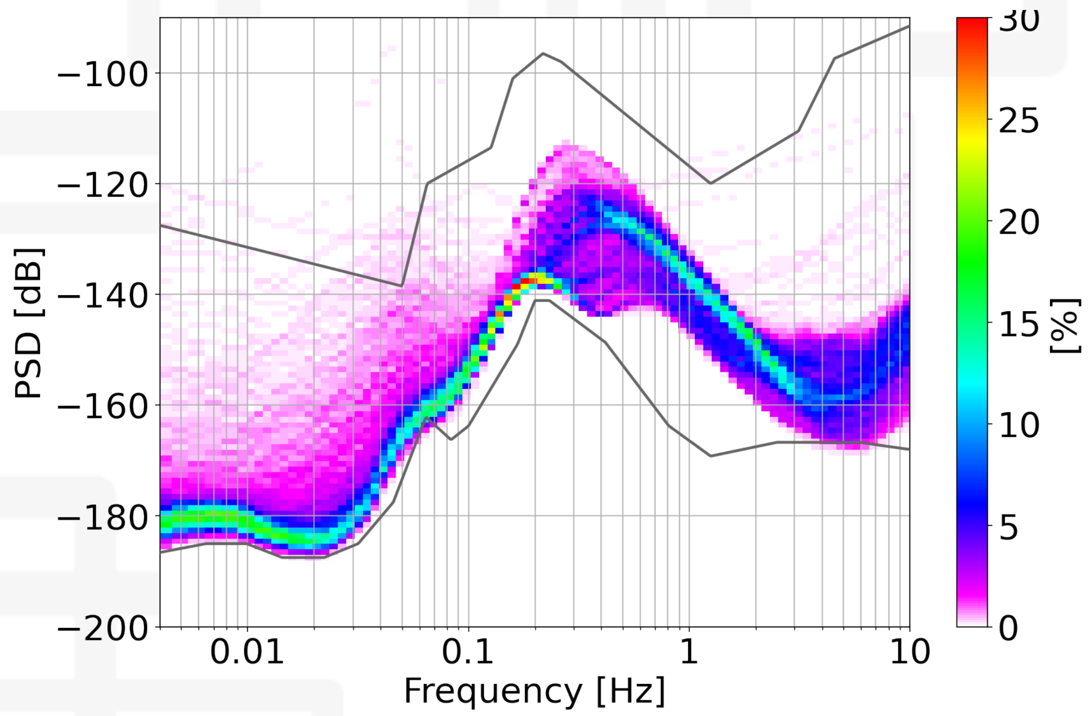
Based on the procedure by McNamara et al. [2004]. Saves and plots PSD histograms.

Also the spectra from the single time segments are saved with their time stamps.

The procedure divides the frequency range into octaves and the in each octave and average is done. This heavily smoothes the spectra.

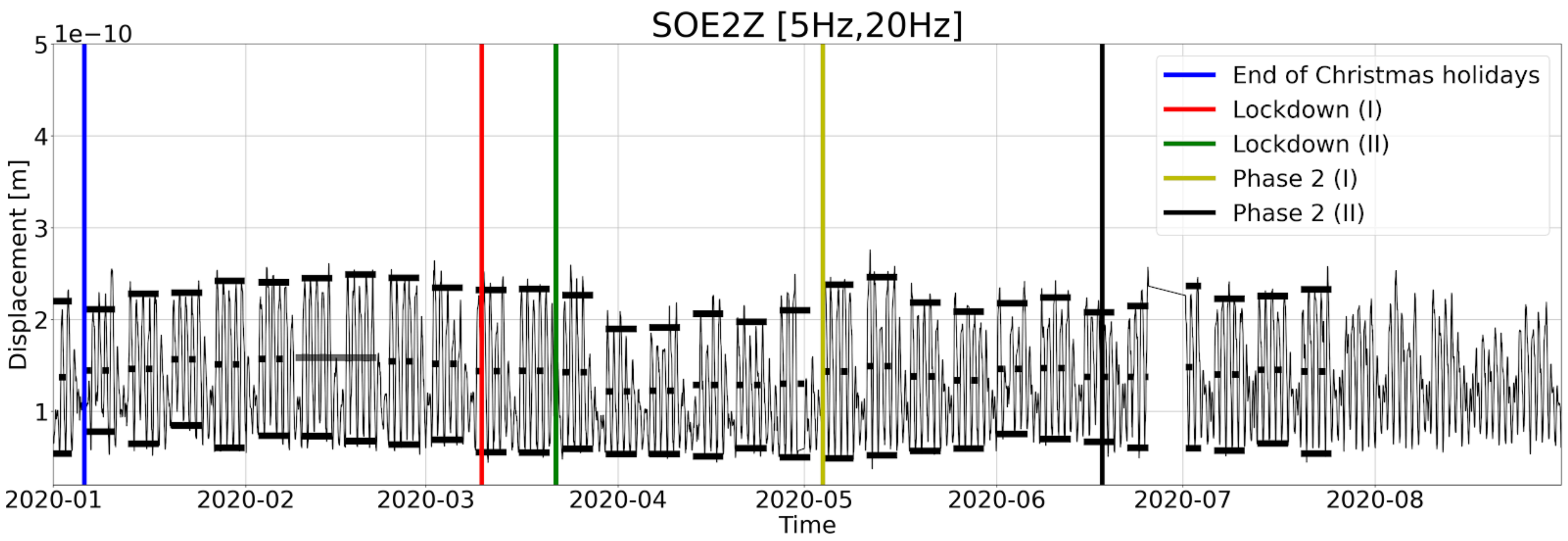
So far I calculated spectra on 1 hr segments with a 50% overlap.

ObsPy PPSD



Using ObsPy, makes very easy also the calculation of the spectra's time series:

```
extract_psd_values(period=T)
```



GS SI

ObsPy PPSD

Pros:

- easy to use method (little coding required);
- saves all relevant quantities;
- easy read/save;
- calculation of time series and spectrograms straightforward;

Cons:

- not flexible (closed function);
- spectra smoothed too much (miss all spectral lines), can get only a general understanding of noise levels;

SciPy functions

Read data with ObsPy (convert Stream and Trace objects in arrays)

Do I need complex spectra?

YES

```
scipy.fft.fft
scipy.fft.rfft
```

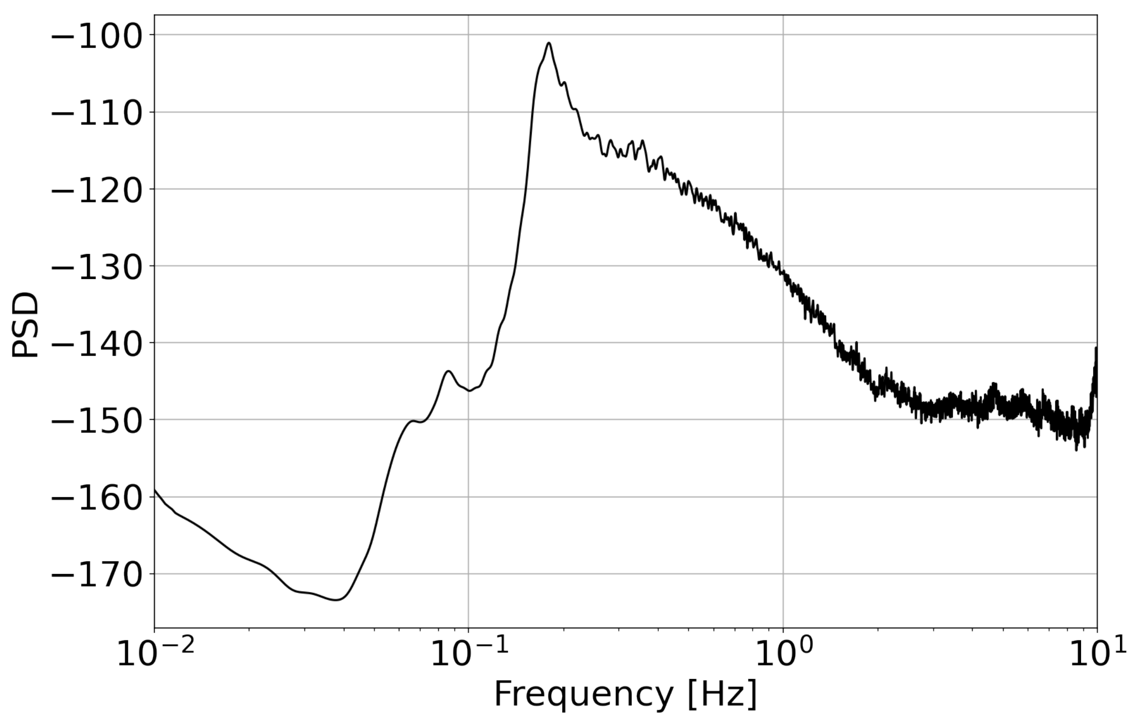
NO

```
scipy.spectrogram
```

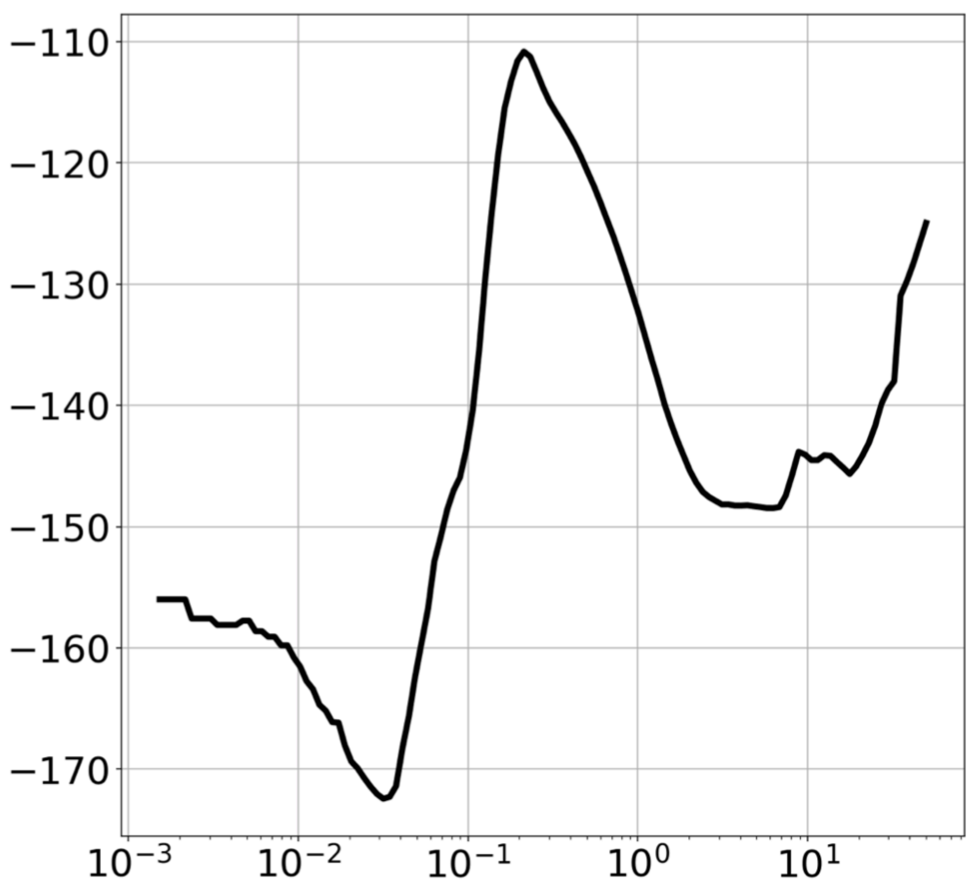
SciPy functions

Compared against ObsPy, the spectra are more full of information

SciPy



ObsPy



SciPy functions

```
scipy.fft.fft
scipy.fft.rfft
```

Pros:

- flexible method;
- reasonably efficient;
- outputs two-sided complex spectra.

Cons:

- Requires more coding;
- calculation of time series cumbersome.

```
scipy.spectrogram
```

Pros:

- handles the division of the data into segments;
- outputs spectrograms;

Cons:

- Real spectra only;
- requires more coding;
- calculation of time series cumbersome.

G S
S I

Things to do with spectra

- Spectral ratios (surface underground):
  - calculated with ObsPy's spectra;
  - 1 hour long segments, 50% overlap;
  - calculated only on spectra segments at same times;
  - output the mean and percentiles.
- H/V ratio:
  - calculated using procedures used in literature (get raw spectra, apply konno-omachi smoothing, make ratios, output mean and percentiles),

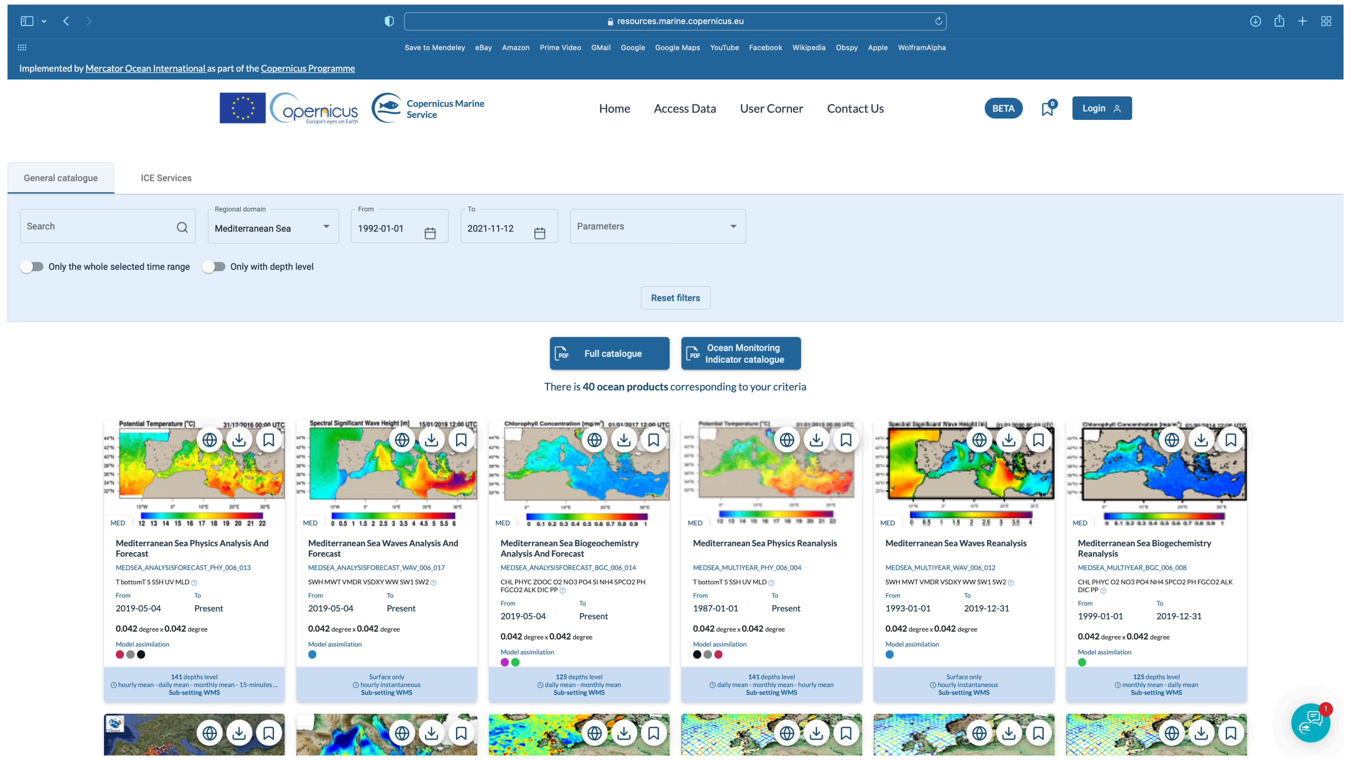
Hands-on material

See the attached notebook and python script for calculating the spectral ratios and the H/V ratio.

Downloading sea wave data

Copernicus is an open data service provided by EU and covers a wide variety of environmental variables including sea data (Copernicus Marine Service).

- get a free account on Copernicus;
- go to https://resources.marine.copernicus.eu/products



Copernicus Marine Service

copernicus Europe's eyes on Earth

Implemented by MERCATOR OCEAN INTERNATIONAL

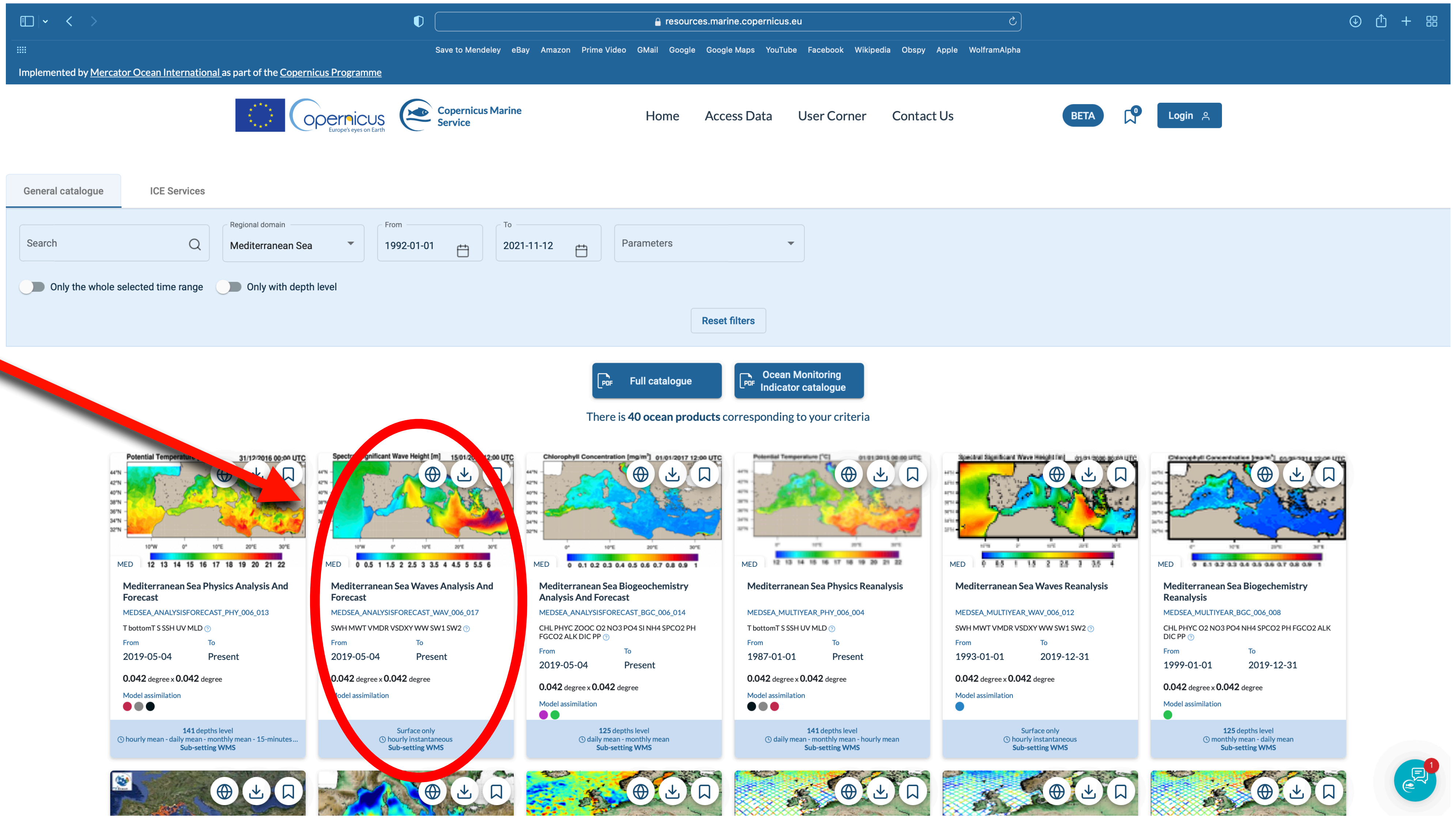
GSSI

Downloading sea wave data

Select "Mediterranean Sea Waves Analysis and Forecast"

Copernicus Marine Service

Implemented by MERCATOR OCEAN INTERNATIONAL



Downloading sea wave data

Go to “Data Select” and choose the following data access method.

You will be prompted to an interface with all the download options. The relevant variable is VHM0.

Downloading sea wave data

Alternatively, you can download the `motuclient` module to download the data through command line interface on your local machine:

```
python -m motuclient --motu http://nrt.cmems-du.eu/motu-web/
Motu --service-id MEDSEA_ANALYSISFORECAST_WAV_006_017-TDS --
product-id med-hcmr-wav-an-fc-h --longitude-min 0 --longitude-
max 17 --latitude-min 35 --latitude-max 45 --date-min "2021-1-1
00:00:00" --date-max "2021-3-31 23:00:00" --variable VMDR_WW --
out-dir="output_directory" --out-name="name_of_file.nc" --user
your_user --pwd your_password
```

Copernicus Marine Service

Implemented by MERCATOR OCEAN INTERNATIONAL

GSSI

Downloading sea wave data

G S
S I

Downloading sea wave data